

Solving the LPN problem in cube-root time *

Urs Wagner

e-mail: urs.wagner@math.uzh.ch

Mathematics Institute

University of Zürich

Winterthurerstr 190, CH-8057 Zürich, Switzerland

January 24, 2012

Abstract

In this paper it is shown that given a sufficient number of (noisy) random binary linear equations, the Learning from Parity with Noise (LPN) problem can be solved in essentially cube root time in the number of unknowns. The techniques used to recover the solution are known from fast correlation attacks on stream ciphers. As in fast correlation attacks, the performance of the algorithm depends on the number of equations given. It is shown that if this number exceeds a certain bound, and the bias of the noisy equations is polynomial in number of unknowns n , the running time of the algorithm is reduced to $2^{\frac{n}{3}+o(n)}$ compared to the brute force checking of all 2^n possible solutions. The mentioned bound is explicitly given and it is further shown that when this bound is exceeded, the complexity of the approach can even be further reduced.

Key Words: LPN, cryptanalysis, fast correlation attack, stream cipher

Subject Classification: 94A60

1 Introduction

In many cryptanalyses, especially in fast correlation attacks on stream ciphers, some information on the secret key is leaked in form of a set of linear binary equations which are satisfied with probability bigger than one half. For each of these equations, let $q = \frac{1}{2} + \epsilon$ be the probability that the secret key is in the solution set. We call ϵ the bias. It is clear that if $\epsilon = \frac{1}{2}$, every equation essentially halves the number of possible solutions, as long as it is independent from the previous ones. In particular if the system of equation has full rank, the key can be recovered in polynomial time by simple Gaussian Elimination. An interesting problem lies in how to recover the key if $\epsilon < \frac{1}{2}$. The LPN problem (e.g. [6],[7]) captures the essence of this task. Let

- $x \in \mathbb{F}_2^n$ be a n -dimensional binary vector, also referred to as the key in the sequel.
- $E \sim \text{Ber}(p)$ be a random variable with $\Pr(E = 1) = p = \frac{1}{2} - \epsilon$ and $\Pr(E = 0) = q = \frac{1}{2} + \epsilon$, $\epsilon \in [0, \frac{1}{2}]$.
- \mathcal{O}_ϵ be an oracle that uniformly at random chooses $g \in \mathbb{F}_2^n$ and outputs pairs $(\langle g, x \rangle + e, g)$ where e is drawn according to E and $\langle \cdot, \cdot \rangle$ denotes the usual inner product. The g 's can be seen as binary linear equations and computing the scalar product with x corresponds to evaluating them at x .

The n -dimensional LPN_ϵ problem can be stated as follows: Given \mathcal{O}_ϵ and ϵ , recover x . A lower bound on the number N of oracle calls necessary in order to be able to identify the correct x with non-negligible probability can be given. This bound corresponds to the number of samples N necessary in order to make a good guess whether a random variable X is distributed according to $\text{Be}(p)$ or $X \sim \text{Be}(\frac{1}{2})$, where Be denotes the usual

*Partially supported by ArmaSuisse funding ARAMIS R3210/047-12 and SNF grant No. 121874.

bernoulli distribution. It is common knowledge that this number satisfies $N = O\left(\frac{1}{\epsilon^2}\right)$, as can easily be seen by Hoeffding's inequality [2]. As a consequence x can be recovered in time $O(2^n \log \frac{1}{\epsilon^2})$ making $N = O\left(\frac{1}{\epsilon^2}\right)$ oracle calls. This is achieved by evaluating all equations at 2^n points, using the techniques of fast Walsh transform [3]. While the LPN problem is proven to be NP-hard [1], in the case where $N \gg \frac{1}{\epsilon^2}$ faster approaches than brute-force checking of all potential keys are possible. Especially the techniques known from fast correlation attacks (see e.g. [3], [4], [8]) are well applicable. The core of most techniques lies in finding linear combinations of the given equations such that a hypothesis on a subset of keybits can be tested. The application of these techniques to the LPN problem has been studied already in e.g. [2, 5, 7]. While the attack we consider is not different to e.g. the one in [5], the approach to the problem is another. From past work, e.g. [2, 3, 5, 4, 7], it is not immediately clear how the complexity behaves depending on the number N of random linear equations and the bias ϵ . The influence of N and ϵ becomes explicit in our considerations. We will show that if $\epsilon = \frac{1}{\text{poly}(n)}$, then the LPN problem can be solved in time $2^{\frac{n}{3} + o(n)}$ given $N = 2^{\frac{n}{\log n} + o(n)}$ equations.

The paper is organized as follows. In Section 2 a short overview on the fast correlation attack techniques is given. In Section 3 it is shown how the complexity to recover the secret key depends on the number N of oracle calls and the main result is stated at the end of the section. Section 3.1 contains the case where the number N of given equations exceeds the bound sufficient for a cube root attack. In Section 4 an illustrating example is given. Throughout the paper, \log will denote the logarithm to base 2.

2 Linear Combination and Hypothesis Testing

Most fast correlation attacks rely on the principles of linear combination and hypothesis testing. The goal of linear combination lies in constructing binary linear equations that depend only on a subset of the keybits. These equations can then be used to test a hypothesis on this subset of keybits. Let $g'_i = (\langle g_i, x \rangle + e_i, g_i) \in \mathbb{F}_2^{n+1}$ be a sample output by the oracle \mathcal{O}_ϵ . Note that if we add w random samples $g'_{i_1}, \dots, g'_{i_w}$ from \mathcal{O}_ϵ , i.e. if we consider $\tilde{g} = \left(\sum_j \langle g_{i_j}, x \rangle + \sum_j e_{i_j}, \sum_j g_{i_j}\right) = \left(\langle \sum_j g_{i_j}, x \rangle + \sum_j e_{i_j}, \sum_j g_{i_j}\right)$ this looks like a sample output from $\mathcal{O}_{\tilde{\epsilon}}$, with $\tilde{\epsilon} = 2^{w-1}\epsilon^w$. This can be seen by the well known Piling-up lemma (e.g. [10]). By appropriately choosing w -tuples of samples from \mathcal{O}_ϵ , we can get equations from $\mathcal{O}_{\tilde{\epsilon}}$ which depend only on a subset of keybits.

Lemma 1. *Let $w \in \mathbb{N}$ be even and $N \gg w$ be the number of samples given from \mathcal{O}_ϵ . Then all w -ary linear combinations of these equations which are all zero in the last b bits can be found in time and space*

$$O\left(\max\left\{N^{\frac{w}{2}}, \frac{N^w}{2^b}\right\}\right). \quad (1)$$

Proof. The number of all $\frac{w}{2}$ -ary linear combinations of the given equations equals

$$\binom{N}{w/2} = \Theta\left(N^{\frac{w}{2}}\right), \quad (2)$$

for fixed w . Compute these linear combinations and store the resulting equations in blocks according to the last b bits, i.e. inside a block the new equations coincide on the last b bits. In each of the 2^b blocks there are an expected number of

$$\frac{\binom{N}{w/2}}{2^b}$$

equations. Inside each block, take all 2-ary combinations which every time gives an expected number of

$$\left(\frac{\binom{N}{w/2}}{2^b}\right)^2$$

equations of the desired form. As there are 2^b blocks, we get an expected number

$$2^b \left(\frac{\binom{N}{w/2}}{2^b}\right)^2 = \Theta\left(\frac{N^{w/2}}{2^b}\right), \quad (3)$$

equations of the desired form. The complexity of the whole is the sum of the complexities for getting all the $\frac{w}{2}$ -ary linear combinations (2) and all the 2-ary combinations (3) inside the 2^b blocks. \square

Clearly the $\frac{N^w}{2^b}$ equations found as in Lemma 1 depend on the first $n - b$ keybits only. A hypothesis on these bits can be tested if

$$\frac{N^w}{2^b} \geq \frac{1}{\epsilon^{r2}} = \frac{1}{2^{2(w-1)\epsilon^{2w}}}, \quad (4)$$

as discussed in Section 1. Note that we implicitly assume that the new equations are pairwise independent, what seems to be a admissible assumption [8]. In order to find the correct $n - b$ keybits, all possible hypotheses on these bits are checked. This can be done by techniques of the fast Walsh transform [3].

Lemma 2. *Evaluating $\frac{1}{2^{2(w-1)\epsilon^{2w}}}$ binary linear equations in $n - b$ variables can be done in time*

$$2^{n-b} \log \frac{1}{2^{2(w-1)\epsilon^{2w}}}. \quad (5)$$

In the next section we derive the optimal choice of the parameters w and b , and we will see how the resulting complexity behaves depending on N .

3 Cube-root algorithm

Suppose we are given $N \geq \frac{1}{\epsilon^2}$ samples from \mathcal{O}_ϵ . In Section 2 we have seen that

- if for $w, b \in \mathbb{N}$ it holds that w is even and

$$\frac{N^w}{2^b} \geq \frac{1}{2^{2(w-1)\epsilon^{2w}}}, \quad (6)$$

then we can recover the first $n - b$ keybits in time

$$O \left(\max \left\{ N^{\frac{w}{2}}, \frac{N^w}{2^b}, 2^{n-b} \log \frac{1}{2^{2(w-1)\epsilon^{2w}}} \right\} \right). \quad (7)$$

In this section we will show how to find optimal parameters b and w such that the expression in (7) is minimal under the condition that the inequality (6) is satisfied. Clearly (6) is equivalent to

$$w (\log N + 2 + 2 \log \epsilon) \geq b + 2,$$

by taking the logarithm on both sides. We will now show that in order to reach minimal complexity in (7) this inequality must be satisfied with equality. Note that the right hand side of the inequality is increasing with b and as $N \geq \frac{1}{\epsilon^2}$, the left hand side is increasing with w . Suppose that for a given choice of b and w the inequality is strict. Then either b can be increased or w can be decreased resulting in a decrease of the overall complexity (7), while the inequality still holds. So we can require equality

$$w (\log N + 2 + 2 \log \epsilon) - 2 = b. \quad (8)$$

Using this in equation (7), we get the following overall complexity

$$O \left(\max \left\{ N^{\frac{w}{2}}, \frac{1}{2^{2(w-1)\epsilon^{2w}}}, 2^n \frac{1}{N^w} \frac{1}{2^{2(w-1)\epsilon^{2w}}} \log \frac{1}{2^{2(w-1)\epsilon^{2w}}} \right\} \right).$$

In order to ease discussion we adjust the condition on N . From now on we will assume that

$$N \geq \frac{4}{(2\epsilon)^4}.$$

As a direct consequence

$$N^{\frac{w}{2}} \geq \frac{1}{2^{2(w-1)\epsilon^{2w}}},$$

and the overall complexity equals

$$O \left(\max \left\{ \underbrace{N^{\frac{w}{2}}}_{\alpha(w)}, \underbrace{2^n \frac{1}{N^w} \frac{1}{2^{2(w-1)} \epsilon^{2w}} \log \frac{1}{2^{2(w-1)} \epsilon^{2w}}}_{\beta(w)} \right\} \right).$$

One readily verifies that $\alpha(w)$ is growing with w and $\beta(w)$ is decreasing with w . Hence the whole term reaches its minimum at the intersection of the two functions, i.e. if $\alpha(w) = \beta(w)$. In order to get an (approximate) solution for the equation $\alpha(w) = \beta(w)$, we ignore the logarithmic term in $\beta(w)$ and obtain:

$$w = \frac{n+2}{3/2 \log N + 2 + 2 \log \epsilon}. \quad (9)$$

Using (8), for b we obtain:

$$b = w (\log N + 2 + 2 \log \epsilon) - 2 = n - \frac{(n+2) \log N}{3 \log N + 4 + 4 \log \epsilon} = n - \frac{w}{2} \log N. \quad (10)$$

We will now examine how this choice of the parameters affects the complexity of the linear combination and hypothesis testing approach. For simplicity in the further analysis let us define

$$T_\epsilon(N) := \frac{\log N}{3 \log N + 4 + 4 \log \epsilon}. \quad (11)$$

So we can write

$$w = \frac{2(n+2)}{\log N} T_\epsilon(N), \quad (12)$$

and

$$b = w (\log N + 2 + 2 \log \epsilon) - 2 = n - (n+2) T_\epsilon(N). \quad (13)$$

Lemma 3. *Notation as in the considerations before. Making $N \geq \frac{4}{(2\epsilon)^4}$ oracle calls and writing $r := w - 2 \lfloor \frac{w+1}{2} \rfloor$ and $T := T_\epsilon(N)$, the n -dimensional LPN_ϵ problem can be solved in time and space*

$$O \left(2^{(n+2)T + |r| \log N + \log((n+2)T + \log N)} \right). \quad (14)$$

Proof. Let w be as in (9). Define

$$w' := 2 \left\lfloor \frac{w+1}{2} \right\rfloor \text{ and } b' := \lfloor w' (\log N + 2 + 2 \log \epsilon) - 2 \rfloor. \quad (15)$$

This definition ensures that w' and b' are integers and $w' = w + r$ is even with $r \in [-1, 1]$. Further

$$\frac{N^{w'}}{2^{b'}} \geq \frac{N^w}{2^b} = \frac{1}{2^{2(w-1)} \epsilon^{2w}},$$

so we have enough equations to check a hypothesis on the $n - b'$ nonzero bits. The complexity for finding the w' -ary linear equations equals

$$N^{\frac{w'}{2}} = N^{\frac{w}{2} + \frac{r}{2}} = 2^{(n+2)T + \frac{r}{2} \log N}.$$

Let us now examine the complexity for evaluating these equations at $2^{n-b'}$ points. We have

$$\begin{aligned} b' &= \lfloor w' (\log N + 2 + 2 \log \epsilon) - 2 \rfloor \\ &= \lfloor (w+r) (\log N + 2 + 2 \log \epsilon) - 2 \rfloor \\ &= \lfloor w (\log N + 2 + 2 \log \epsilon) - 2 + r (\log N + 2 + 2 \log \epsilon) \rfloor \\ &\geq w (\log N + 2 + 2 \log \epsilon) - 2 + r (\log N + 2 + 2 \log \epsilon) - 1/2 \\ &\stackrel{(13)}{=} n - (n+2)T + r (\log N + 2 + 2 \log \epsilon) - 1/2. \end{aligned}$$

Hence

$$2^{n-b'} \log \frac{N^{w'}}{2^{b'}} \leq 2^{(n+2)T-r(\log N+2+2\log \epsilon)+1/2} \underbrace{\log N^{\frac{w'}{2}}}_{\leq (n+2)T+\frac{1}{2}\log N}.$$

Adding these two upper bounds, we obtain the overall complexity

$$2^{n-b'} \log \frac{N^{w'}}{2^{b'}} + N^{\frac{w'}{2}} \leq 2^{(n+2)T} \left(2^{-r(\log N+2+2\log \epsilon)+\frac{1}{2}+\log((n+2)T+\frac{r}{2}\log N)} + 2^{\frac{r}{2}\log N} \right) \quad (16)$$

$$= O \left(2^{(n+2)T+r\log N+\log((n+2)T+\frac{1}{2}\log N)} \right) \quad (17)$$

□

Corollary 1. *Using the notation from the previous lemma. Making $N \geq \frac{4}{(2\epsilon)^4}$ oracle calls the n -dimensional LPN_ϵ problem can be solved in time and space*

$$O \left(2^{(n+2)T+\log N+\log((n+2)T+\log N)} \right).$$

Proof. Immediately as $|r| = |w - 2 \lfloor \frac{w+1}{2} \rfloor| \leq 1$. □

It is not hard to see that if $\log N$ is large, $T_\epsilon(N)$ will converge to $\frac{1}{3}$. Clearly

$$T_\epsilon(N) = \frac{\log N}{3\log N + 4 + 4\log \epsilon} = \frac{1}{3} \left(1 - \frac{4 + 4\log \epsilon}{3\log N + 4 + 4\log \epsilon} \right).$$

Recall that $\log \epsilon < -1$ and since $N > \frac{4}{(2\epsilon)^4}$ we have that $\log N > -2 - 4\log \epsilon$. Consequently

$$T_\epsilon(N) < \frac{1}{3} \left(1 + \frac{4 + 4\log \epsilon}{2 + 8\log \epsilon} \right) < \frac{1}{3} \left(1 + \frac{1}{2} \right) = \frac{1}{2}.$$

In the case where N is significantly bigger, particularly if

$$N \geq \frac{4}{(2\epsilon)^4} 2^{\frac{n}{\log n}} > 2^{\frac{n}{\log n} - \frac{4}{3} - \frac{4}{3}\log \epsilon},$$

one readily verifies that

$$T_\epsilon(N) \leq \frac{1}{3} \left(1 - \frac{4(1 + \log \epsilon) \log n}{n} \right). \quad (18)$$

We can prove the following lemma:

Lemma 4. *If $\epsilon = \frac{1}{\text{poly}(n)}$ we can solve the n -dimensional LPN_ϵ in time and space*

$$2^{\frac{n}{3}+o(n)},$$

making $N \geq 2^{\frac{n}{\log n}} \frac{4}{(2\epsilon)^4}$ oracle calls.

Proof. First notice that with $\epsilon = \frac{1}{\text{poly}(n)}$, we have that $\log \frac{1}{\epsilon} = o(n)$. We will use only a subset of $N' = 2^{\frac{n}{\log n}} \frac{4}{(2\epsilon)^4}$ of the given equations. Write $T' := T_\epsilon(N')$. Then

$$(n+2)T' = \frac{n+2}{3} \left(1 - \frac{4\log n(\log \epsilon + 1)}{3n} \right) = \frac{1}{3}n + \frac{2}{3} - \underbrace{\frac{4(n+2)}{n}(\log \epsilon + 1)\log n}_{=o(n)}.$$

Further

$$\log N' = \frac{n}{\log n} + \log \frac{4}{(2\epsilon)^4} = o(n).$$

So from Lemma 3 and as also $\log((n+2)T' + \log N') = o(n)$, we get that we can find the solution in time

$$O \left(2^{(n+2)T'+\log N'+\log((n+2)T'+\log N')} \right) = 2^{\frac{n}{3}+o(n)}.$$

□

We have seen that we do not need more than $N = 2^{\frac{n}{\log n} + o(n)}$ equations to solve the LPN problem in essentially cube-root time. As seen in the proof of Lemma 4, given $N \gg 2^{\frac{n}{\log n}}$ the approach makes use of $N' = 2^{\frac{n}{\log n} \frac{4}{(2\epsilon)^4}}$ of the given equations. The resulting overhead can be exploited to further reduce the complexity. The principle used in the case where $N \gg 2^{\frac{n}{\log n}}$ is called decimation [5]. Given

$$N = 2^l 2^{\frac{n}{\log n}} \frac{4}{(2\epsilon)^4}$$

equations, the problem is basically reduced to solving the LPN problem in dimension $n - l$.

3.1 Decimation

We have seen that if $N = 2^{\frac{n}{\log n} \frac{4}{(2\epsilon)^4}}$ the complexity of the LPN problem is $2^{\frac{n}{3} + o(n)}$. If we are given $N \gg 2^{\frac{n}{\log n} \frac{4}{(2\epsilon)^4}}$ equations, simple decimation allows to reduce the security parameter n of the problem. Suppose we are given

$$N = 2^{\frac{n}{\log n} + l} \frac{4}{(2\epsilon)^4},$$

equations with $l < n - 2$. We want to consider only the equations that do not depend on (e.g. the first) $l' \in \mathbb{N}$ bits of the key x . We have an expected number $N2^{-l'}$ of such equations. In order to be able to recover the remaining $n - l'$ keybits, the following equality must hold

$$N2^{-l'} = 2^{\frac{n}{\log n} + l - l'} \frac{4}{(2\epsilon)^4} \geq 2^{\frac{n - l'}{\log(n - l')}} \frac{4}{(2\epsilon)^4}.$$

Equivalently,

$$\frac{n}{\log n} + l - l' \geq \frac{n - l'}{\log(n - l')}.$$

Setting $l' = \lfloor l \rfloor \leq n - 3$, this inequality is satisfied and we can reduce the problem parameter n to $n - l$.

Lemma 5. *If $\epsilon = \frac{1}{\text{poly}(n)}$ we can solve the LPN_ϵ in time and space*

$$2^{\frac{n - l}{3} + o(n)}.$$

making $N \geq 2^{\frac{n}{\log n} + l} \frac{4}{(2\epsilon)^4}$ oracle calls.

4 Example

We have seen that the LPN problem can be solved in essentially cube-root time and space. Consider the classical setting of a fast correlation attack [9]. Suppose we have a stream cipher with keylength $n = 128$ whose output bits correspond to linear combinations of the keybits transmitted over the binary symmetric channel with crossover probability $\frac{1}{2} - \epsilon = \frac{1}{2} - \frac{1}{8} = 0.375$. For a given number $N \geq 2^{10}$ (note that $\log \frac{4}{(2\epsilon)^4} = 10$) of equations, we have seen how to in principle optimally choose w and b (see (9) and (10)). However these values are not necessarily in \mathbb{N} and w is not necessarily even. So w is rounded to the nearest even number w' and b' is chosen accordingly (see (15) in the proof of Lemma 3). This gives an additional summand $< |r| \log N$ in the exponent of the complexity (compare (14)). Table 4 shows how this rounding problem influences the complexity. Decimation is not considered in this example.

Acknowledgment

The author would like to thank G. Maze for many useful comments and discussions.

$\log N$	w	b	w'	b'	$ r \log N$	$\log C_{LC}$	$\log C_{HT}$
10	11.82	68.91	12	70	1.8	60	63.64
20	5	78	6	94	20	60	38.70
30	3.17	80.44	4	102	24.8	60	30.17
40	2.32	81.57	2	70	12.8	40	61.32
47	1.95	82.06	2	84	2.1	47	47.32
50	1.83	82.22	2	90	8.5	50	41.32

Table 1: Complexity of Linear Combination C_{LC} and Hypothesis testing C_{HT} depending on the number of equations.

References

- [1] E. Berlekamp, R. McEliece, and H. Van Tilborg. On the inherent intractability of certain coding problems. *IEEE Transactions on Information Theory*, 24, 1978.
- [2] A. Blum, A. Kalai, and H. Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *J.ACM*, 50.
- [3] P. Chose, A. Joux, and M. Mitton. Fast correlation attacks: An algorithmic point of view. In *Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, 2002, Proceedings*, pages 209–221, 2002.
- [4] M. Fossorier, M. Mihaljevic, and H. Imai. Modeling block decoding approaches for the fast correlation attack. *IEEE Trans. Inform. Theory*, 54(12):4728–4737, 2007.
- [5] M. Fossorier, M. Mihaljevic, H. Imai, Y. Cui, and K. Matsuura. A novel algorithm for solving the LPN problem and its application to security evaluation of the HB protocol for RFID authentication. volume 4329 of *Lecture Notes in Computer Science*, pages 48–62. Springer, 2006.
- [6] A. Juels and S. Weis. Authenticating pervasive devices with human protocols. In *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*, pages 293–308. Springer-Verlag, 2005.
- [7] E. Levieil and P. Fouque. An Improved LPN Algorithm. In *Security and Cryptography for Networks, 5th International Conference, SCN 2006*, volume 4116 of *Lecture Notes in Computer Science*, pages 348–359, Maiori, Italy, 2006. Springer.
- [8] P. Lu and L. Huang. A new correlation attack on lfsr sequences with high error tolerance. In *Coding, Cryptography and Combinatorics*, volume 23 of *Progress in Computer Science and Applied Logic*, pages 67–83. Birkhauser, Basel, 2004.
- [9] W. Meier and O. Staffelbach. Fast correlation attack on stream ciphers. In *Advances in cryptology—EUROCRYPT '88*, volume 330 of *Lecture Notes in Comput. Sci.*, pages 301–316. Springer, Berlin, 1988.
- [10] S. Vaudenay. *A Classical Introduction to Cryptography: Applications for Communications Security*. Springer Verlag, 2006.